

The CVT Interface Library

Copyright 1999-2006, United States Government as represented by the Administrator of the National Aeronautics and Space Administration. No copyright is claimed in the United States under Title 17, U.S. Code.

This software and documentation are controlled exports and may only be released to U.S. Citizens and appropriate Permanent Residents in the United States. If you have any questions with respect to this constraint contact the GSFC center export administrator, <Thomas.R.Weisz@nasa.gov>.

This product contains software from the Integrated Test and Operations System (ITOS), a satellite ground data system developed at the Goddard Space Flight Center in Greenbelt MD. See <<http://itos.gsfc.nasa.gov>> or e-mail <itos@itos.gsfc.nasa.gov> for additional information.

You may use this software for any purpose provided you agree to the following terms and conditions:

1. Redistributions of source code must retain the above copyright notice and this list of conditions.
2. Redistributions in binary form must reproduce the above copyright notice and this list of conditions in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product contains software from the Integrated Test and Operations System (ITOS), a satellite ground data system developed at the Goddard Space Flight Center in Greenbelt MD.

This software is provided "as is" without any warranty of any kind, either express, implied, or statutory, including, but not limited to, any warranty that the software will conform to specification, any implied warranties of merchantability, fitness for a particular purpose, and freedom from infringement and any warranty that the documentation will conform to their program or will be error free.

In no event shall NASA be liable for any damages, including, but not limited to, direct, indirect, special or consequential damages, arising out of, resulting from, or in any way connected with this software, whether or not based upon warranty, contract, tort, or otherwise, whether or not injury was sustained by persons or property or otherwise, and whether or not loss was sustained from or arose out of the results of, or use of, their software or services provided hereunder.

1 Introduction

The functions in the CVT Interface Library provide a higher level interface to the dbif functions.

Imagine a application that needs to increment a global mnemonic. Using the CVT Interface Library, the application can increment the mnemonic via

```
cvtSetVInt(NULL, "GBL_INCREMENT", cvtGetVInt(NULL, "GBL_INCREMENT") + 1);
```

This is considerably more concise and readable than the alternative:

```
Tmnem tmnem;
...
if (!dbTmnemOK(tmnem=dbTmLookupName("GBL_INCREMENT"))) {
    EvtMsg(CFG_ERROR, "GBL_INCREMENT is not a mnemonic!\n");
} else {
    TmValue *tmv;
    if ((tmv=dbTmGetValue(NULL, tmnem)) == NULL) {
        EvtMsg(TCWFAULT, "dbTmGetValue() failed -- %s\n", liberrmsg);
    } else {
        int i = *tmv->value.i + 1;
        if (dbTmSetValue(tmnem, &i, NULL) != 0) {
            EvtMsg(TCWFAULT, "dbTmSetValue() failed -- %s\n", liberrmsg);
        }
        dbFree(tmv);
    }
}
```

2 Functions

**CVTIF * cvtifInit (CVTIF *buf, verbose_handler, error_handler,
panic_handler, malloc_handler)** Function

cvtifInit initializes the *CVTIF ** argument to the other CVT Interface library functions. For convenience, *cvtifInit* is not required to be called; if it isn't called, NULL should be passed as the *CVTIF ** parameter to the other functions.

The parameters to *cvtifInit* are functions that glue the CVT Interface library to the rest of the application.

If *buf* is NULL, a CVTIF structure will be allocated using *malloc_handler* or *malloc()*.

void verbose_handler (char *format. ...) Function

This function gets called to produce verbose output, and is intended to be used only while debugging. If NULL, verbose output is not produced.

void error_handler (char *format. ...) Function

This function gets called to produce error messages. If NULL, the following function is used:

```
static void
default_error_handler(char *format, ...)
{
    va_list argp;
    va_start(argp, format);
    vEvtMsg(CFG_ERROR,format,argp);
}
```

void panic_handler (char *format, ...) Function

This function gets called to produce panic messages. If NULL, the following function is used:

```
static void
default_panic_handler(char *format, ...)
{
    va_list argp;
    va_start(argp, format);
    vEvtMsg(CFG_ERROR,format,argp);
}
```

void malloc_handler (char *format, ...) Function

This function gets called to allocate memory. If NULL, *malloc()* is used.

void cvtifTerm (CVTIF cvtif) Function

cvtifTerm terminates the *cvtif* package and deallocates any resources associated with *cvtif*.

```
void CVTIF_ERROR_HANDLER (CVTIF *cvtif) (char *format,  
...)  
void CVTIF_PANIC_HANDLER (CVTIF *cvtif) (char *format,  
...)
```

These are actually macros, but they allow your application to access the error and panic message handlers. Because these are macros, the usage looks a bit unusual:

```
CVTIF_ERROR_HANDLER(cvtif)("While doing %s saw %d; expected %d!\n",  
task, x, expected_x);
```

int cvtGetVInt (CVTIF *cvtif, char *mnemonic)	Function
double cvtGetVFloat (CVTIF *cvtif, char *mnemonic)	Function
char * cvtGetString (CVTIF *cvtif, char *mnemonic)	Function
CUC_TIME cvtGetVTime42 (CVTIF *cvtif, char *mnemonic)	Function
unsigned cvtGetVUnsigned (CVTIF *cvtif, char *mnemonic)	Function
double cvtGetVAnalog (CVTIF *cvtif, char *mnemonic)	Function
char * cvtGetVDiscrete (CVTIF *cvtif, char *mnemonic)	Function

These functions return a mnemonic's raw or converted value, given the mnemonic's name.

The values returned by *cvtGetString* or *cvtGetVDiscrete* were allocated using *malloc_handler* (or *malloc()*, if *malloc_handler* is NULL) and should be free'd when no longer needed.

int cvtGetTVInt (CVTIF *cvtif, Tmnem tmnem)	Function
double cvtGetTVFloat (CVTIF *cvtif, Tmnem tmnem)	Function
char * cvtGetString (CVTIF *cvtif, Tmnem tmnem)	Function
CUC_TIME cvtGetVTime42 (CVTIF *cvtif, Tmnem tmnem)	Function
unsigned cvtGetTVUnsigned (CVTIF *cvtif, Tmnem tmnem)	Function
double cvtGetTVAnalog (CVTIF *cvtif, Tmnem tmnem)	Function
char * cvtGetTVDiscrete (CVTIF *cvtif, Tmnem tmnem)	Function

These functions return a mnemonic's raw or converted value, given the mnemonic's *tmnem* (see *dbifLookupName()*).

The values returned by *cvtGetString* or *cvtGetTVDiscrete* were allocated using *malloc_handler* (or *malloc()*, if *malloc_handler* is NULL) and should be free'd when no longer needed.

void cvtSetVInt (CVTIF *cvtif, char *mnemonic, int v)	Function
void cvtSetVFloat (CVTIF *cvtif, char *mnemonic, double v)	Function
void cvtSetVString (CVTIF *cvtif, char *mnemonic, char * v)	Function
void cvtSetVTime42 (CVTIF *cvtif, char *mnemonic, CUC_TIME * v)	Function
void cvtSetVUnsigned (CVTIF *cvtif, char *mnemonic, unsigned v)	Function

These functions set a mnemonic's value, given the mnemonic's name and new value.

void cvtSetTVInt (CVTIF *cvtif, Tmnem tmnem, int v)	Function
void cvtSetTVFloat (CVTIF *cvtif, Tmnem tmnem, double v)	Function
void cvtSetTVString (CVTIF *cvtif, Tmnem tmnem, char * v)	Function
void cvtSetTVTime42 (CVTIF *cvtif, Tmnem tmnem, CUC_TIME * v)	Function
void cvtSetTVUnsigned (CVTIF *cvtif, Tmnem tmnem, unsigned v)	Function

These functions set a mnemonic's value, given the mnemonic's *tmnem* (see *dbiLookupName()*) and new value.

Index to functions

C

cvtGetTVAnalog	3
cvtGetTVDiscrete	3
cvtGetTVFloat	3
cvtGetTVInt	3
cvtGetTVString	3
cvtGetTVTime42	3
cvtGetTVUnsigned	3
cvtGetVAnalog	3
cvtGetVDiscrete	3
cvtGetVFloat	3
cvtGetVInt	3
cvtGetVString	3
cvtGetVTime42	3
cvtGetVUnsigned	3
CVTIF_ERROR_HANDLER	3
CVTIF_PANIC_HANDLER	3
cvtifInit	2
cvtifTerm	2
cvtSetTVFloat	4
cvtSetTVInt	3
cvtSetTVString	4
cvtSetTVTime42	4

cvtSetTVUnsigned	4
cvtSetVFloat	3
cvtSetVInt	3
cvtSetVString	3
cvtSetVTime42	3
cvtSetVUnsigned	3

E

error_handler	2
---------------------	---

M

malloc_handler	2
----------------------	---

P

panic_handler	2
---------------------	---

V

verbose_handler	2
-----------------------	---

Table of Contents

1	Introduction	1
2	Functions	2
	Index to functions.....	5